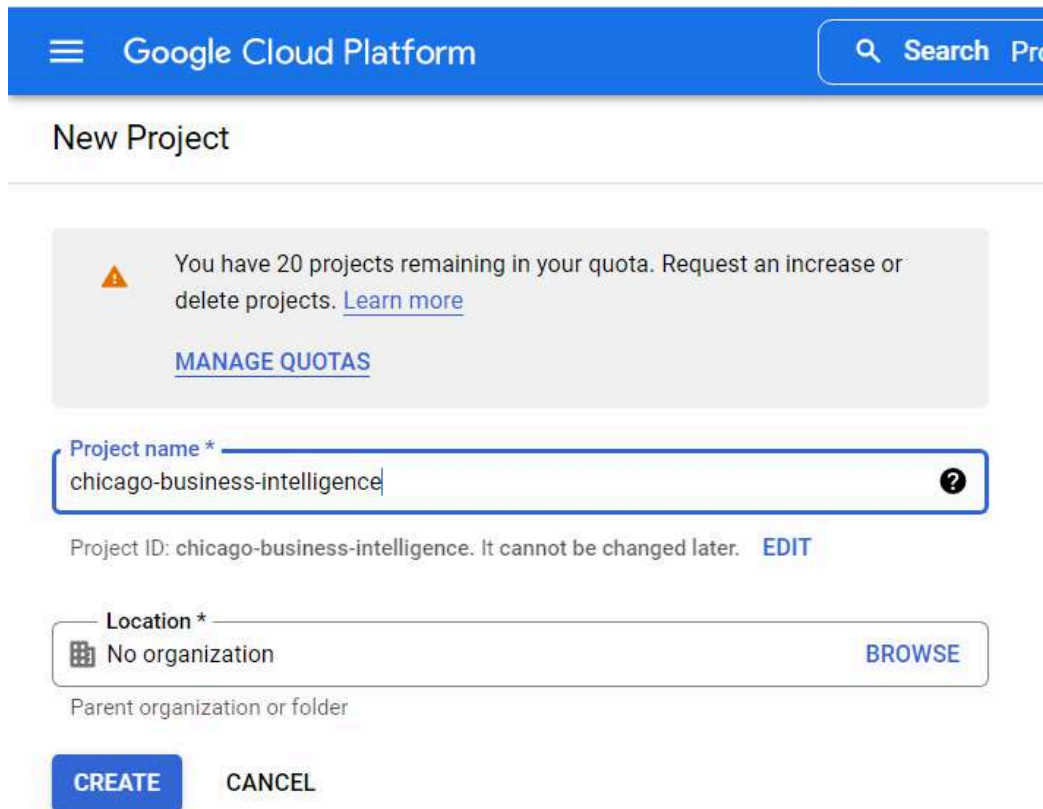


Deploying Go Microservice for Chicago Business Intelligence on GCP using SQL, Cloud Run, CI/CD Triggers for GitHub Repo

Step1: Initial Setup for Google Cloud Platform

- Install the [google cloud CLI](#) on your local machine.
- Create a new project on your [google cloud console](#). Make a note of the project id and project Name.



The screenshot shows the 'New Project' page in the Google Cloud Platform console. At the top is a blue header with the 'Google Cloud Platform' logo and a search bar. Below the header, the title 'New Project' is displayed. A warning box indicates that the user has 20 projects remaining in their quota. The 'Project name' field is filled with 'chicago-business-intelligence'. Below this, the 'Project ID' is shown as 'chicago-business-intelligence'. The 'Location' dropdown is set to 'No organization'. At the bottom, there are 'CREATE' and 'CANCEL' buttons.

Google Cloud Platform

New Project

You have 20 projects remaining in your quota. Request an increase or delete projects. [Learn more](#)

[MANAGE QUOTAS](#)

Project name *

Project ID: chicago-business-intelligence. It cannot be changed later. [EDIT](#)

Location * [BROWSE](#)

Parent organization or folder

[CREATE](#) [CANCEL](#)

- After creating a project on Google Cloud Console execute **"gcloud init"** command on your local machine and select the project created above when prompted.

```
Your current project has been set to: [chicago-business-intelligence].
```

Step 2: Postgres database Setup

- Create a database instance of postgres using the following command.
"gcloud sql instances create mypostgres --database-version=POSTGRES_14 --cpu=2 --memory=7680MB --region=us-central"

```
Command Prompt
C:\Users\user1>gcloud sql instances create mypostgres --database-version=POSTGRES_14 --cpu=2 --memory=7680MB --region=us-central
API [sqladmin.googleapis.com] not enabled on project [753970993858]. Would you like to enable and retry (this will take a few minutes)? (y/N)? y
Enabling service [sqladmin.googleapis.com] on project [753970993858]...
Operation "operations/acat.p2-753970993858-87e383f9-b886-4541-af05-d0243b572649" finished successfully.
Creating Cloud SQL Instance...done.
Created [https://sqladmin.googleapis.com/sql/v1beta4/projects/tensile-medium-358715/instances/mypostgres].
NAME      DATABASE_VERSION  LOCATION    TIER      PRIMARY_ADDRESS  PRIVATE_ADDRESS  STATUS
mypostgres POSTGRES_14       us-central1-f db-custom-2-7680 35.224.24.146    -              RUNNABLE
C:\Users\user1>
```

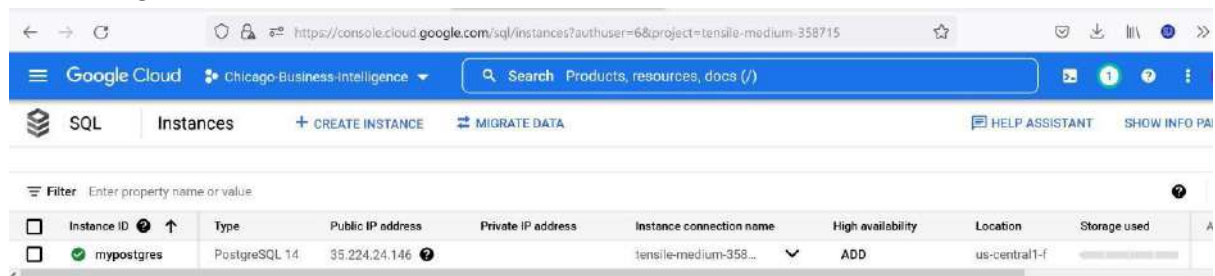
- Create sql users on the database instance using the following command.
"gcloud sql users set-password postgres --instance=mypostgres --password=root"

```
C:\Users\user1>gcloud sql instances create mypostgres --database-version=POSTGRES_14 --cpu=2 --memory=7680MB --region=us-central
API [sqladmin.googleapis.com] not enabled on project [753970993858]. Would you like to enable and retry (this will take a few minutes)? (y/N)? y
Enabling service [sqladmin.googleapis.com] on project [753970993858]...
Operation "operations/acat.p2-753970993858-87e383f9-b886-4541-af05-d0243b572649" finished successfully.
Creating Cloud SQL Instance...done.
Created [https://sqladmin.googleapis.com/sql/v1beta4/projects/tensile-medium-358715/instances/mypostgres].
NAME      DATABASE_VERSION  LOCATION    TIER      PRIMARY_ADDRESS  PRIVATE_ADDRESS  STATUS
mypostgres POSTGRES_14       us-central1-f db-custom-2-7680 35.224.24.146    -              RUNNABLE
C:\Users\user1>
```

- Create a database for our microservice using the following command.
"gcloud sql databases create chicago_business_intelligence --instance=mypostgres"

```
C:\Users\user1>gcloud sql databases create chicago_business_intelligence --instance=mypostgres
Creating Cloud SQL database...done.
Created database [chicago_business_intelligence].
instance: mypostgres
name: chicago_business_intelligence
project: tensile-medium-358715
C:\Users\user1>
```

- Open Google Cloud console, search for SQL and confirm that the database instance is up and running.

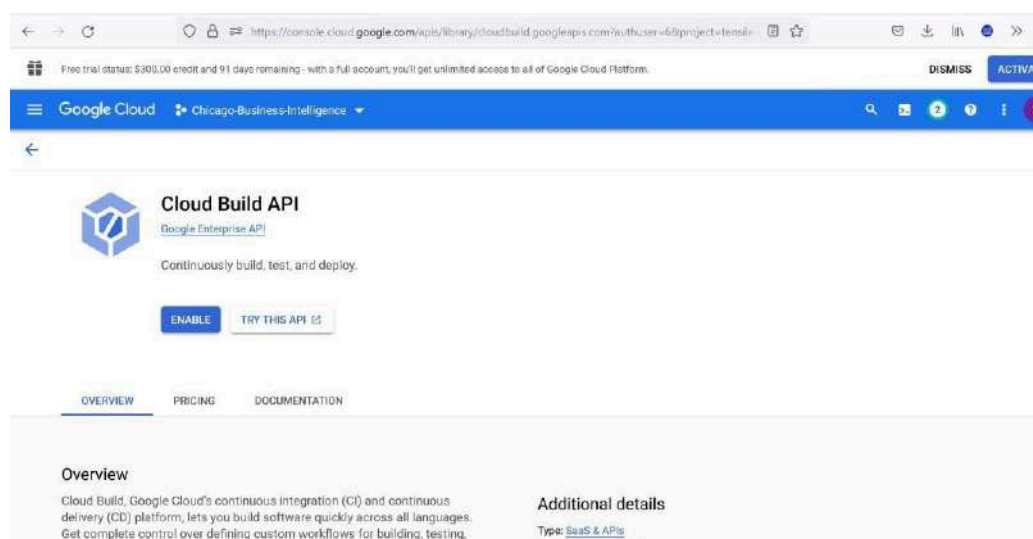
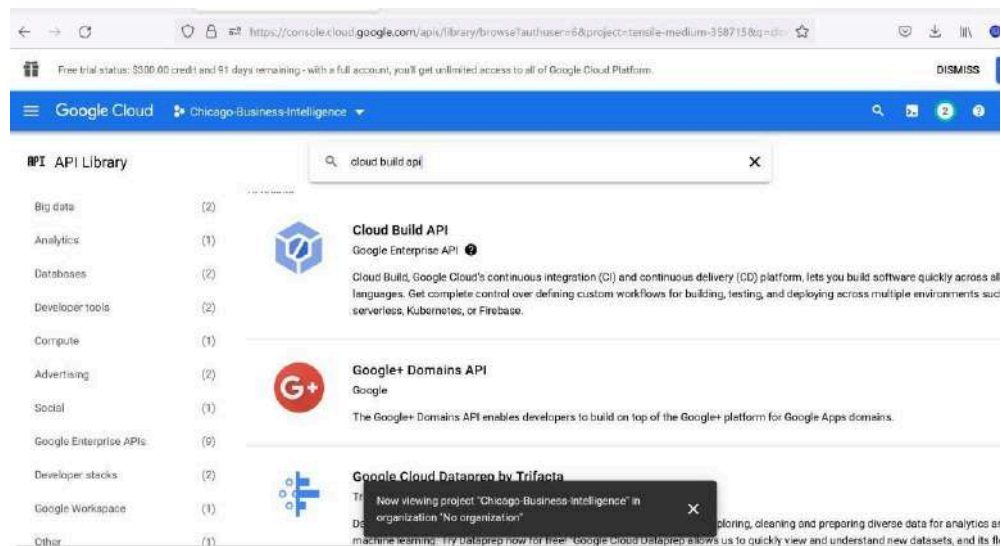


Step 3: Setting up continuous deployment using cloud build.

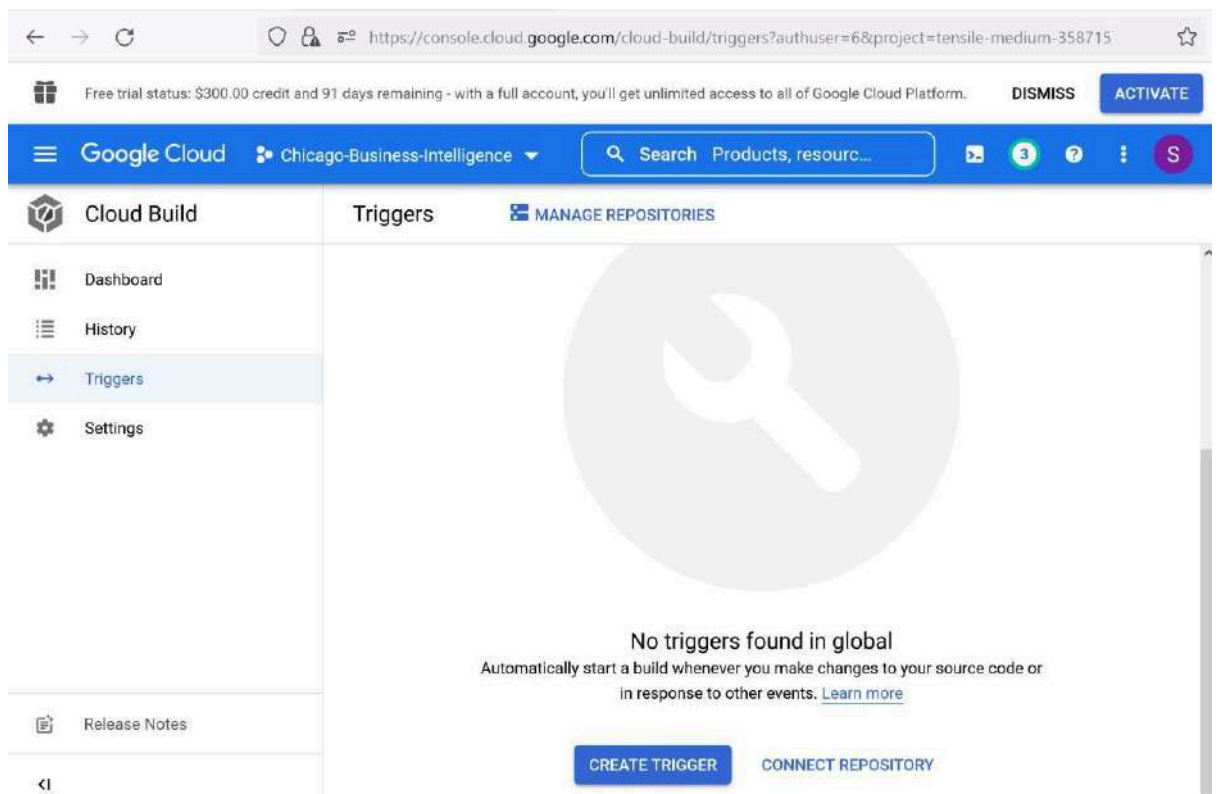
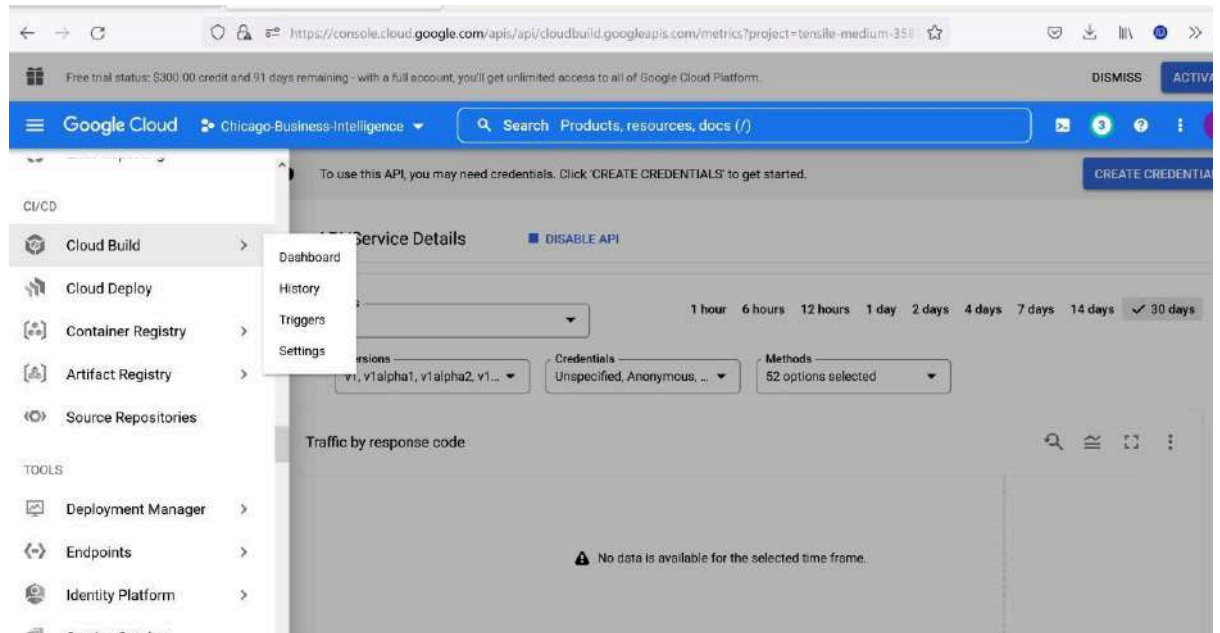
- Create a **Repository** on **GitHub** to store the source code for your CBI project.



- Open Google Cloud Console, Search for Cloud build API and Enable it for your project.



- After the API is enabled, click on the create trigger button.



- Fill the details for the trigger as shown in the below images.

The screenshot shows the 'Create trigger' page in the Google Cloud Console. The left sidebar contains navigation links: Dashboard, History, Triggers (selected), and Settings. The main content area has the following fields:

- Name ***: A text input field containing 'cbi-trigger'. Below it, a note states 'Must be unique within the project's region'.
- Region ***: A dropdown menu set to 'global (non-regional)'.
- Description**: An empty text input field.
- Tags**: An empty text input field with a help icon.
- Event**: A section titled 'Repository event that invokes trigger' with three radio button options:
 - ☒ **Push to a branch**
 - ☐ **Push new tag**
 - ☐ **Pull request** (with a note: 'Not available for Cloud Source Repositories')

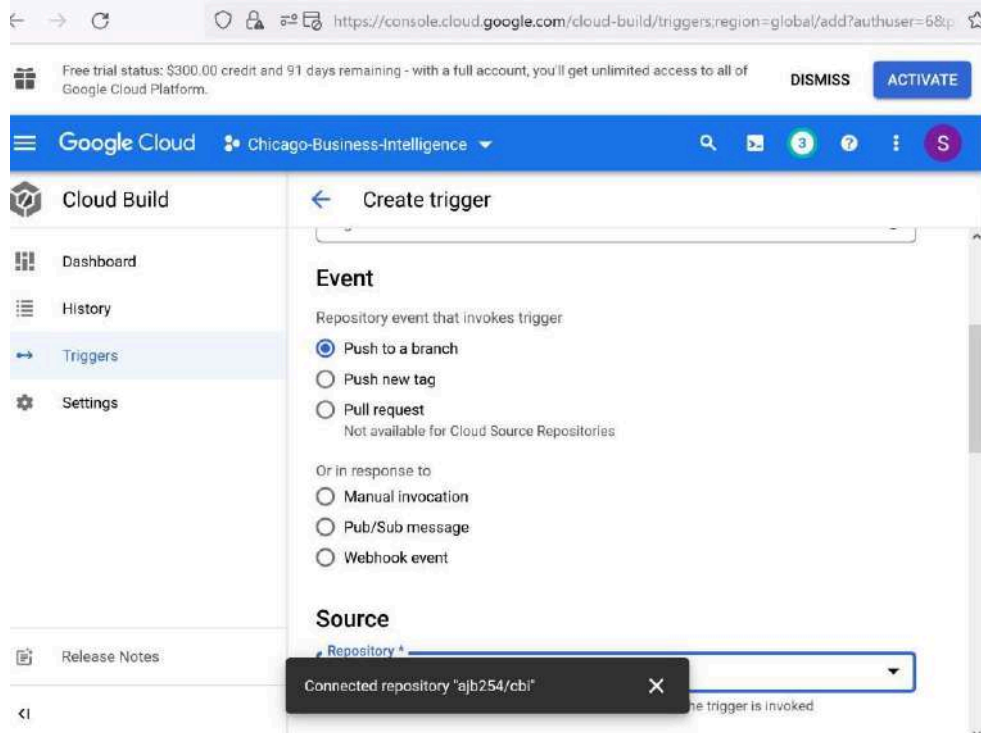
- Click on connect repository, select GitHub and authenticate.

The screenshot shows the 'Connect repository' dialog box overlaid on the 'Create trigger' page. The dialog has two main sections:

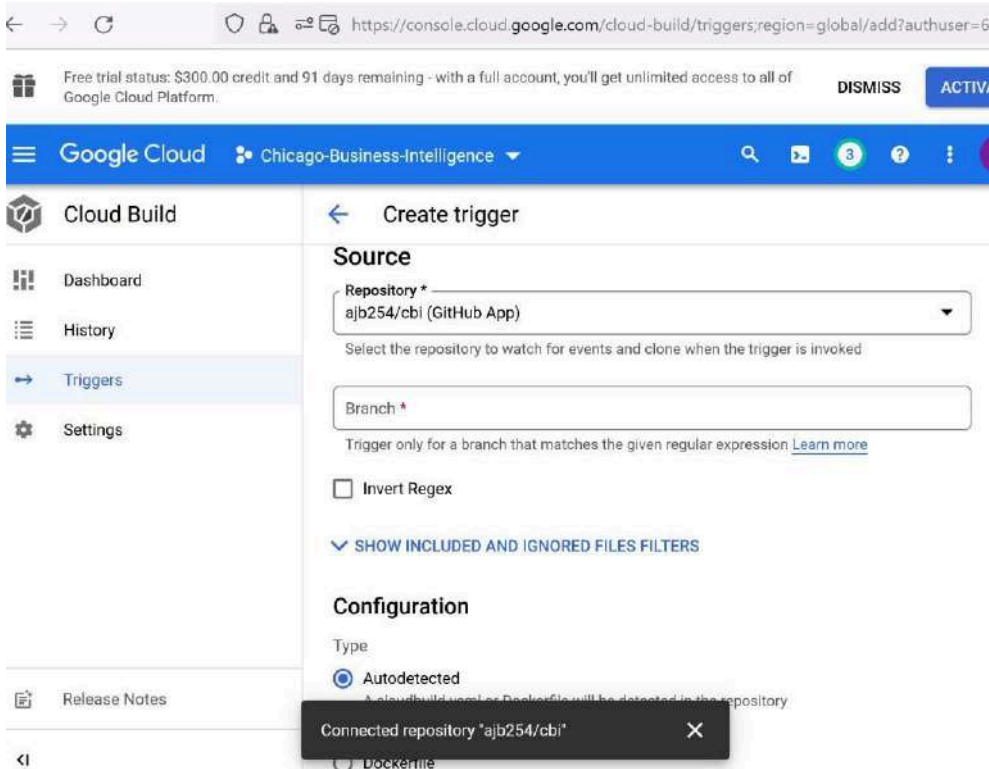
- 1 Select source**: A list of repository providers with radio button selection:
 - ☒ **GitHub (Cloud Build GitHub App)**: Build source code in response to pull requests and pushes.
 - ☐ **GitHub Enterprise**: Build source code hosted on premises in response to pull requests and pushes.
 - ☐ **Bitbucket Server**: Build source code hosted on premises in response to pull requests and pushes.
 - ☐ **Bitbucket Data Center**: Build source code hosted on premises in response to pull requests and pushes.
 - ☐ **Bitbucket Cloud (mirrored)** **BETA**: Build source code in response to pushes, mirrored through Cloud Source Repositories.A 'SHOW MORE' link is visible below the list.
- 2 Authenticate**: A section with the text 'You will be asked to authorize the Google Cloud Build GitHub App to access your GitHub Account to proceed. You may revoke access through GitHub at any time.' and a 'CONTINUE' button.

The background 'Create trigger' page shows the 'Source' section with a 'Repository' field and a 'CONNECT NEW REPOSITORY' button.

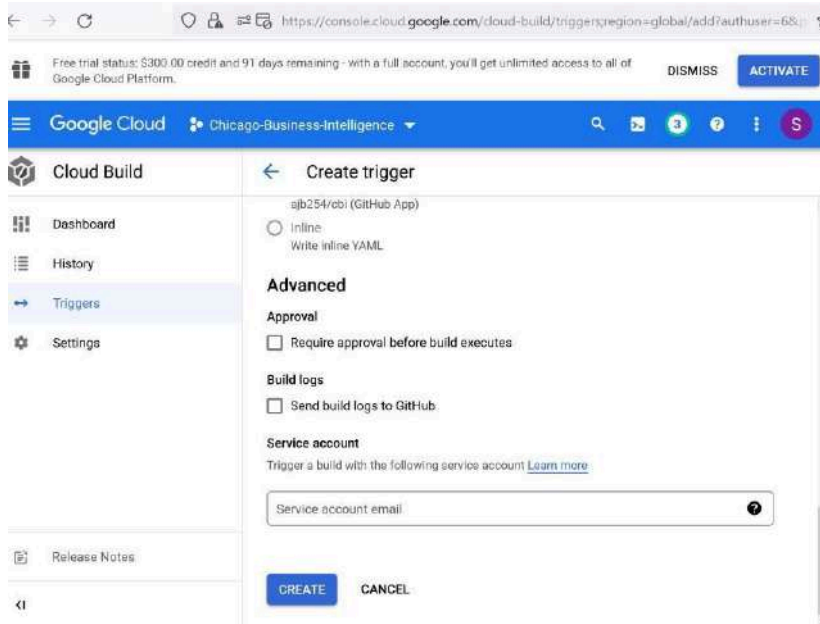
- After authentication select the repository created for Chicago business intelligence.



- Select the repository after connecting the project.

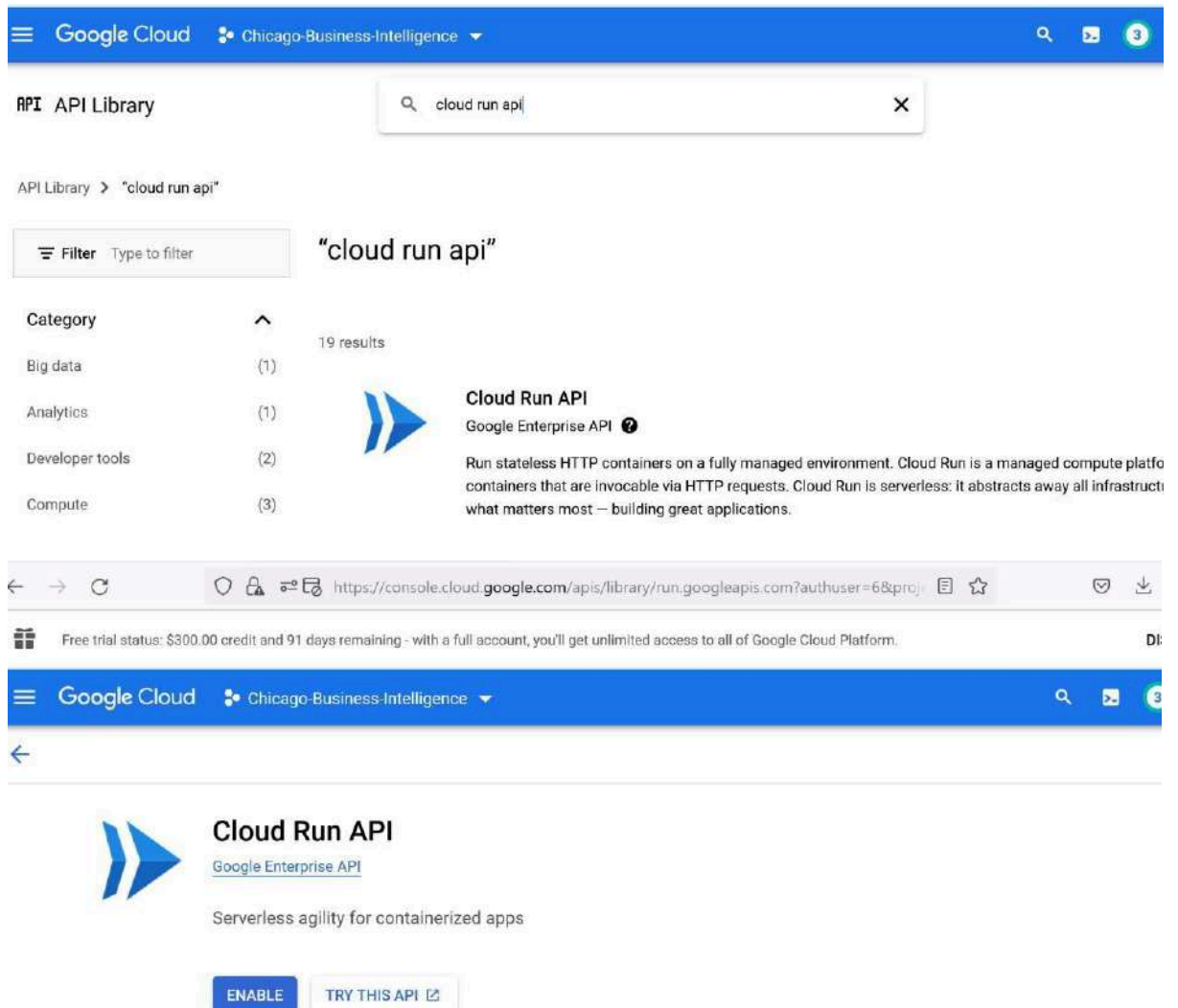


- Click on create to create the trigger.



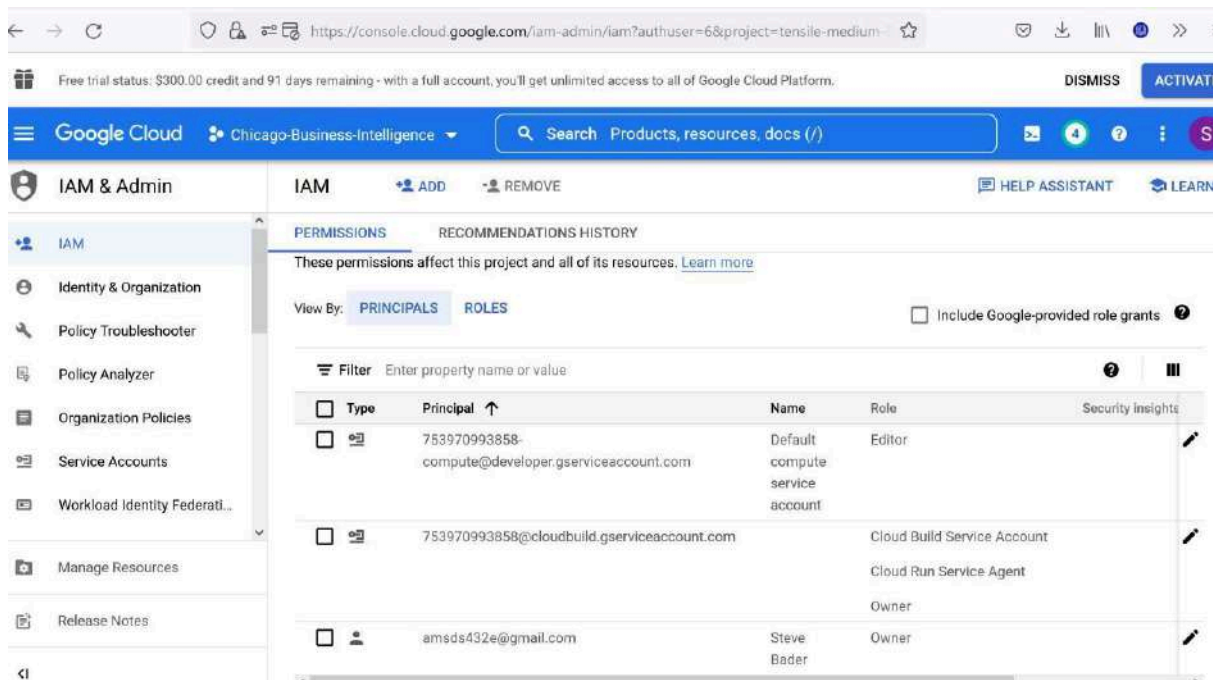
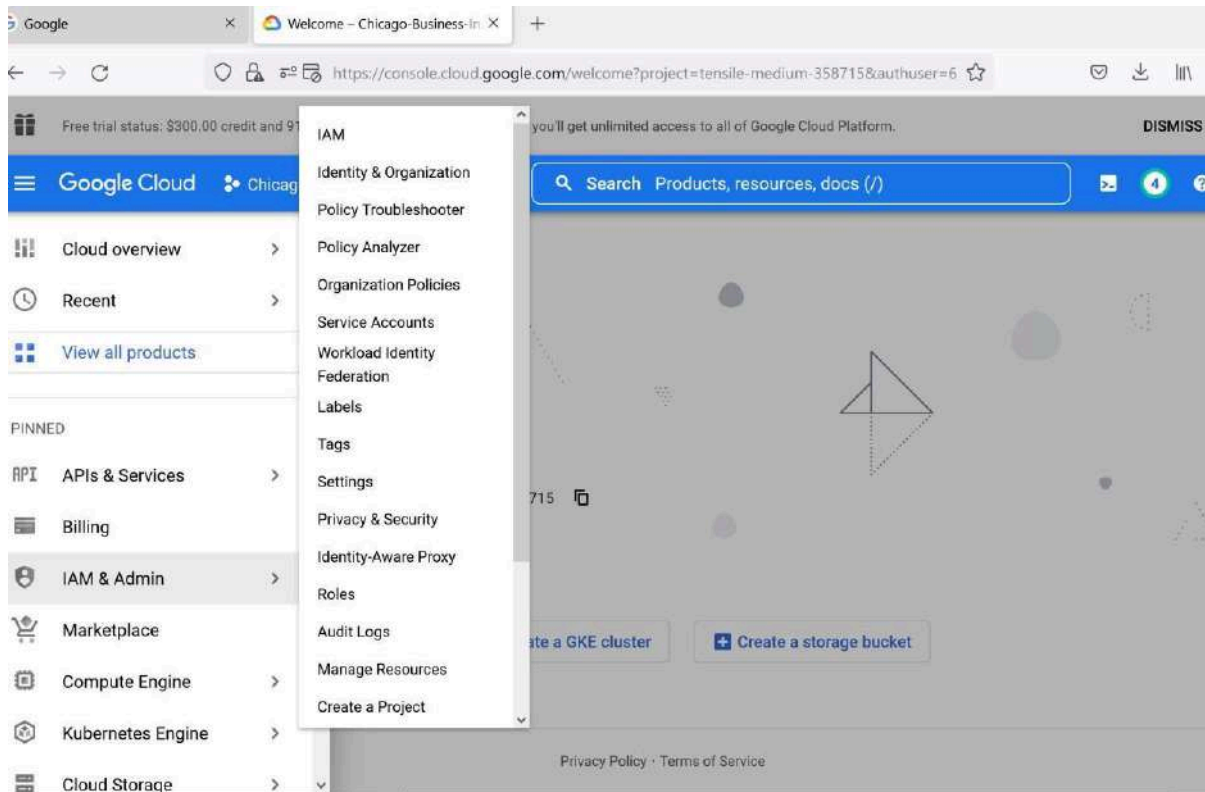
Step 4: Setting up the containers for Go-microservice and Pgadmin

- Enable cloud run api for your project.



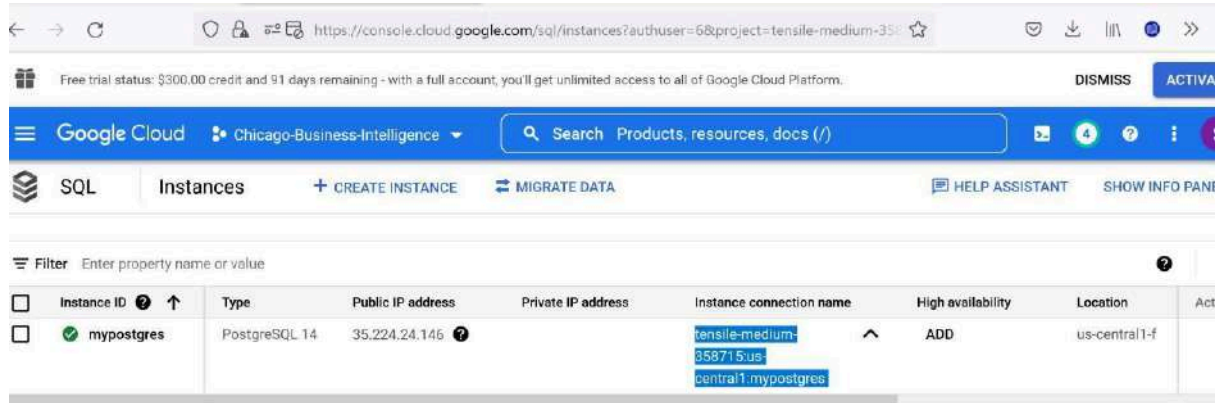
Step 5: Enable IAM permissions/roles

- Go to the IAM page and make sure all the required roles are enabled for the project.



Step 6: Get the Postgres DB instance connection name

- The images for all the microservice are created with the help of cloudbuild.yaml file.
- Go to the postgres instance created in the previous steps and copy the instance connection name.



The screenshot shows the Google Cloud console interface for SQL instances. The breadcrumb navigation indicates the path: Google Cloud > Chicago-Business-Intelligence > SQL > Instances. A table lists the instances, with one instance named 'mypostgres' highlighted. The 'Instance connection name' for this instance is 'tensile-medium-358715-us-central1-mypostgres'.

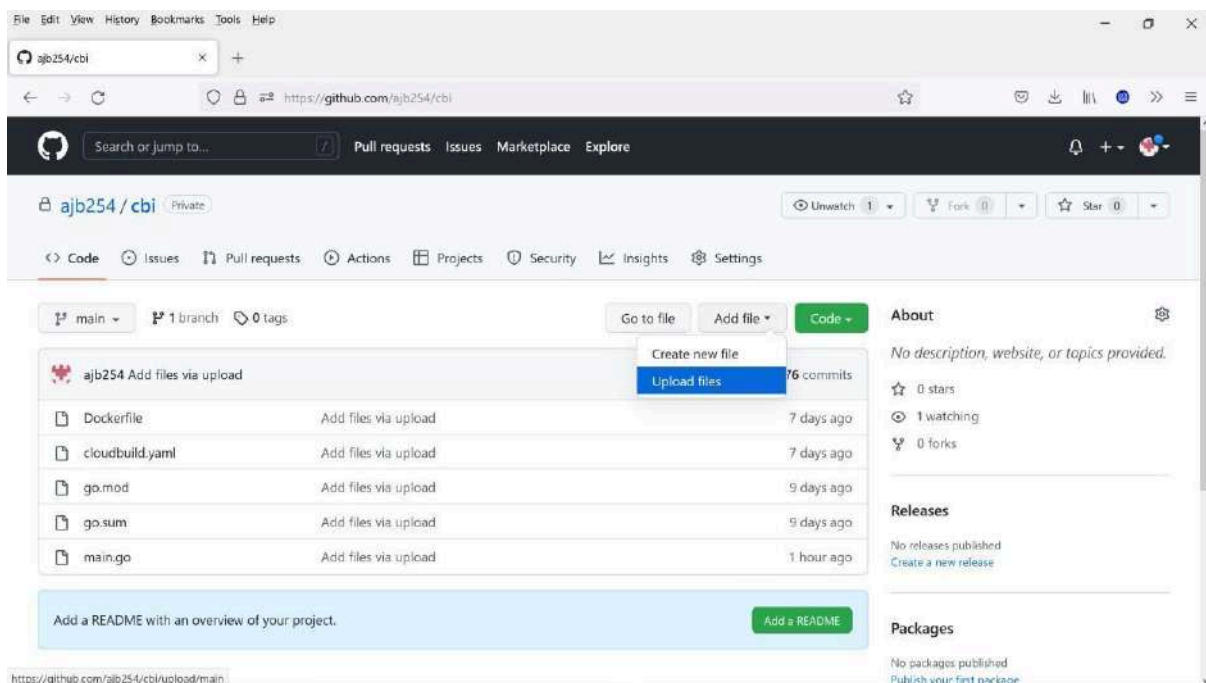
Instance ID	Type	Public IP address	Private IP address	Instance connection name	High availability	Location	Act
✓ mypostgres	PostgreSQL 14	35.224.24.146		tensile-medium-358715-us-central1-mypostgres	ADD	us-central1-f	

- Go to line “.env” file and update the POSTGRES_HOST name string with your instance connection name as shown below. Example is shown below.

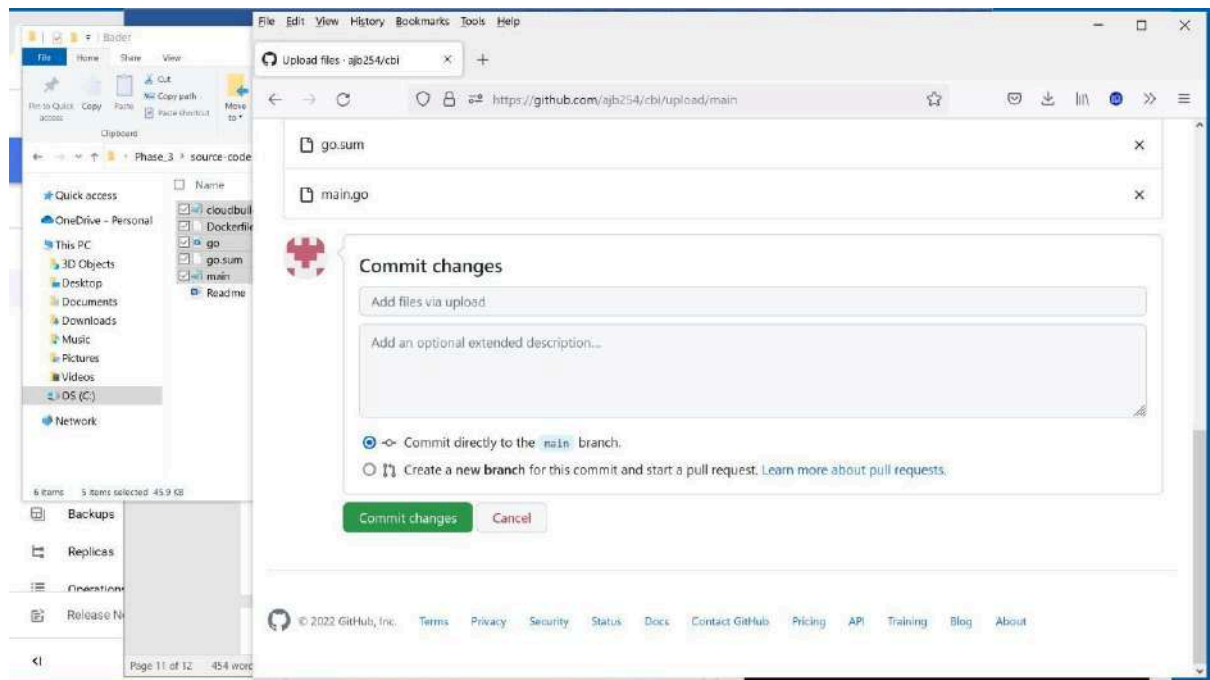
.env

POSTGRES_HOST="/cloudsql/chicago-business-intelligence:us-central1:mypostgres"

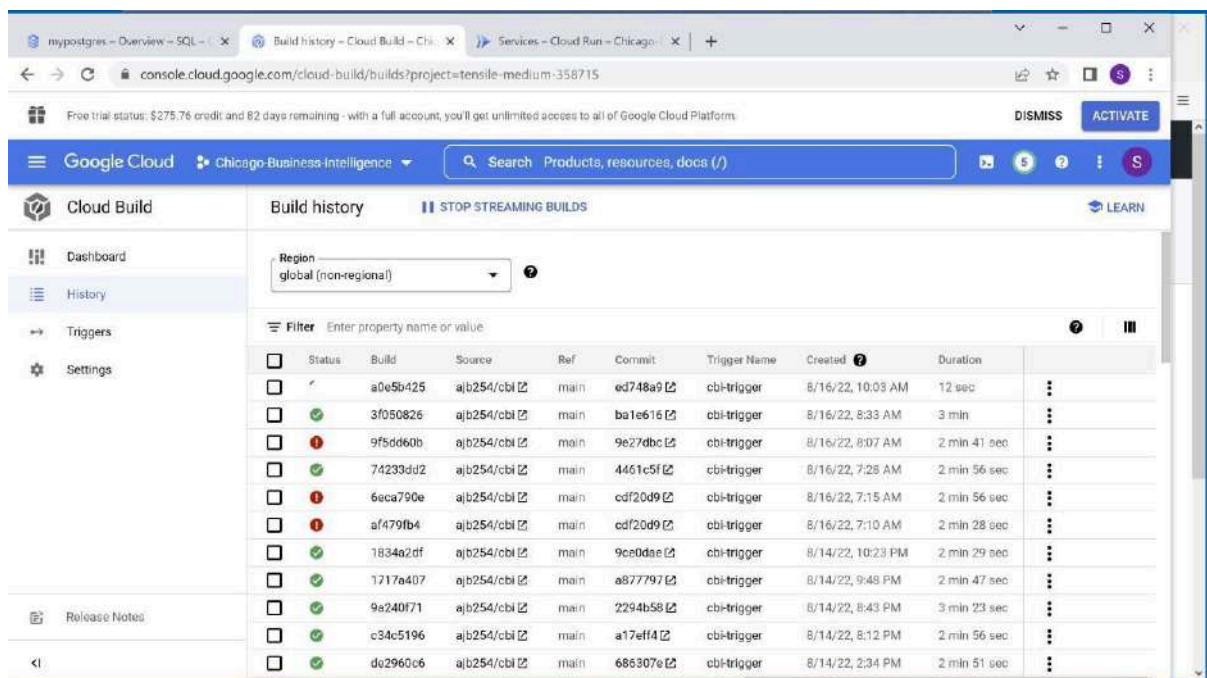
- Push the source code along with the cloudbuild.yaml file to the GitHub repository created in prior steps.



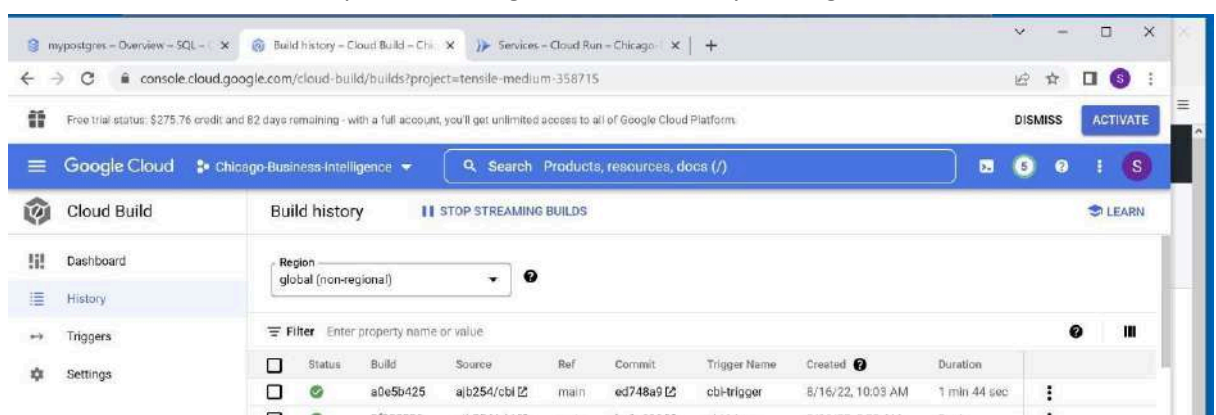
Step 7: Push commits to GitHub, run trigger, and view microservices

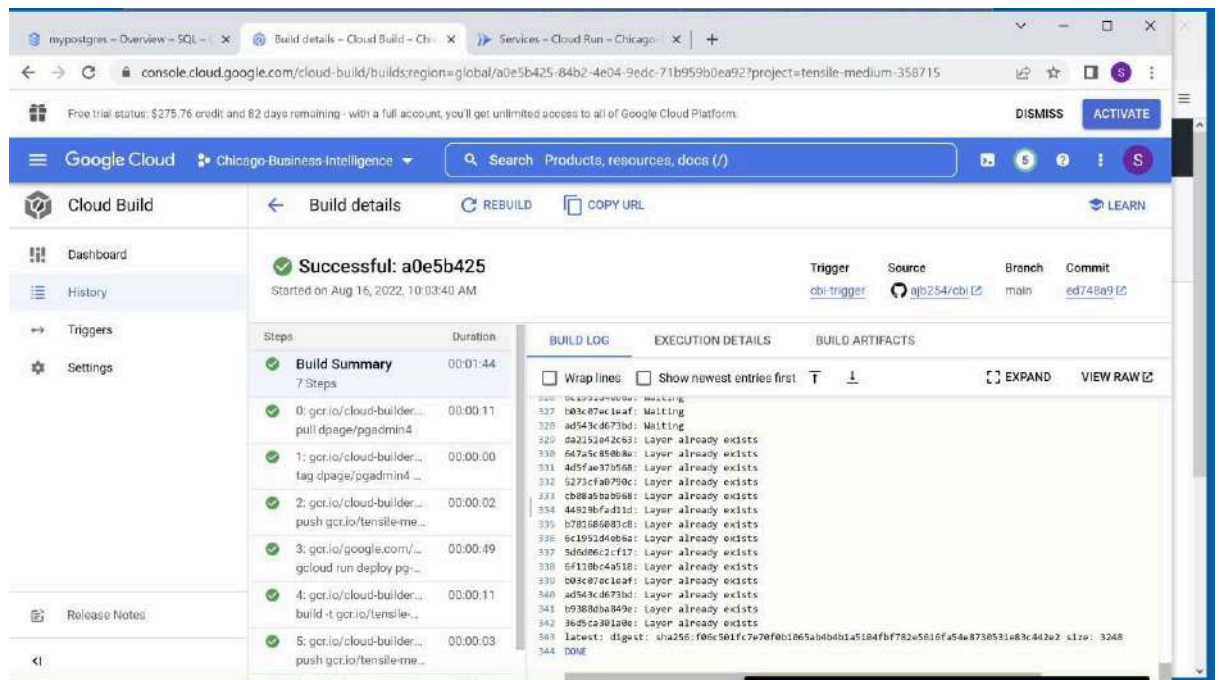


- A build is triggered in cloud build immediately after pushing the code to GitHub.

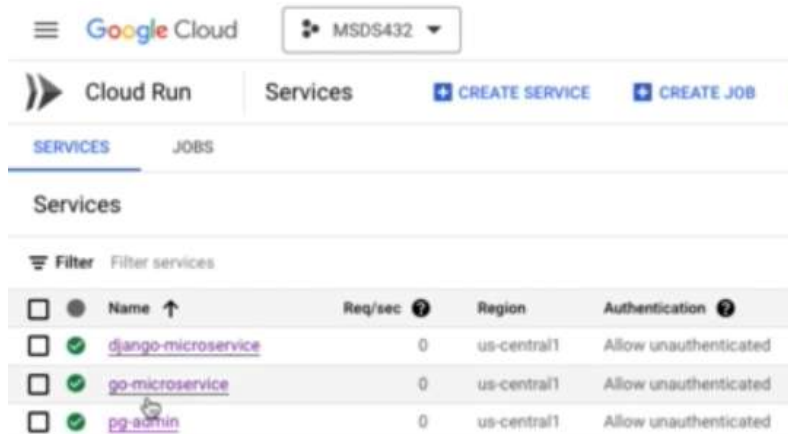


- Wait for the build to be complete. Build logs can be viewed by clicking on the build id.



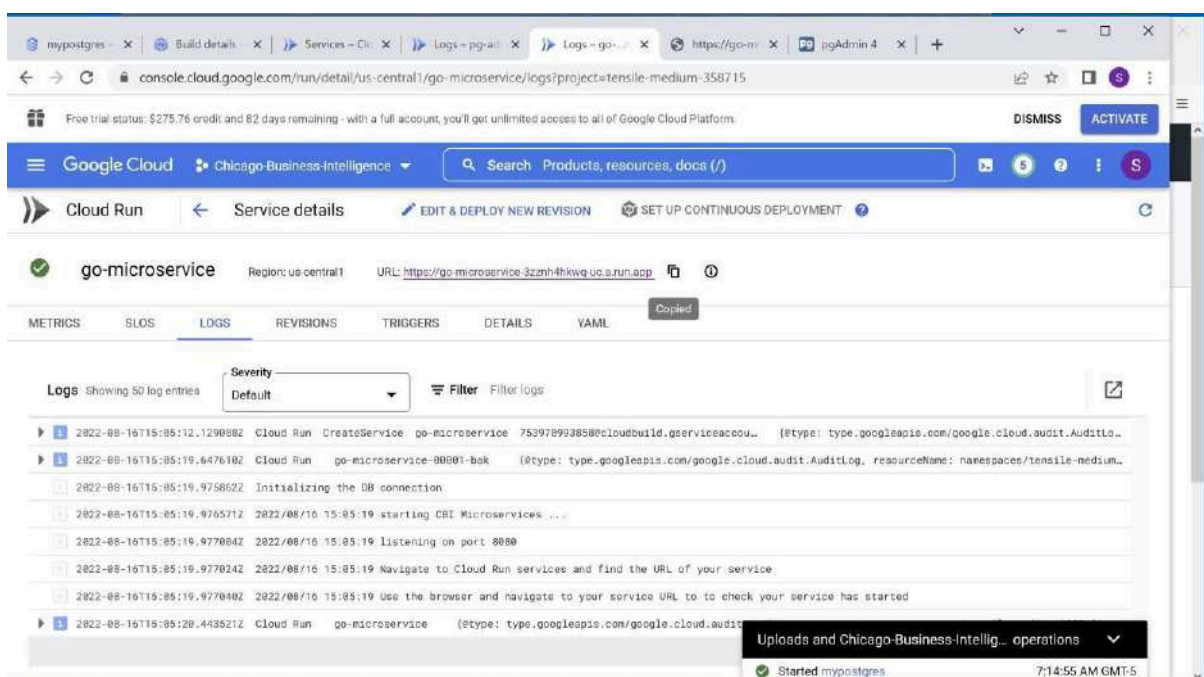


- Go to Cloud Run and Verify you see your services are up and running (green).



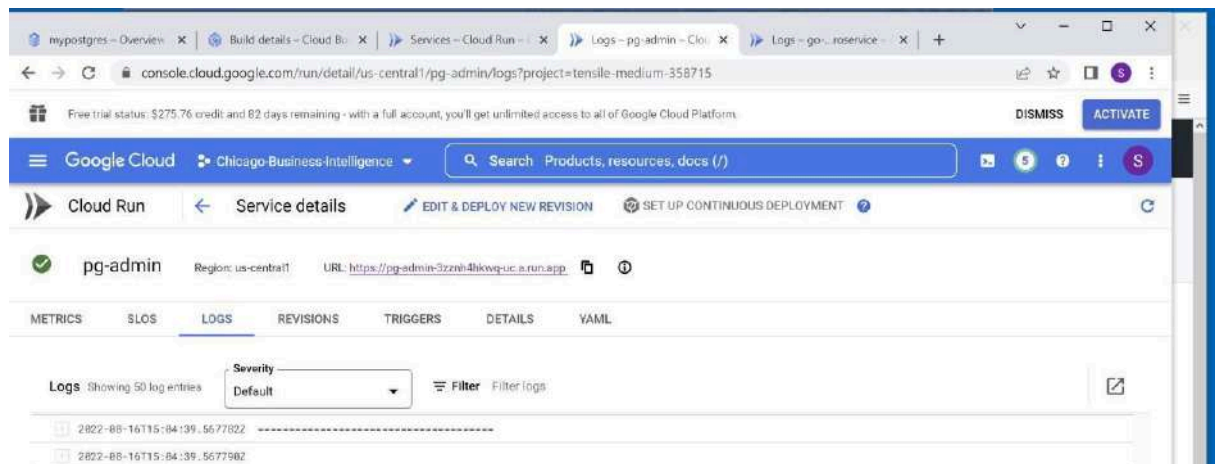
Go Microservice (Backend)

- From Cloud Run, click on the go-microservice app, copy the highlighted URL.
- Open the go-microservice URL in a new browser window and you should see the goroutines for the microservices started.

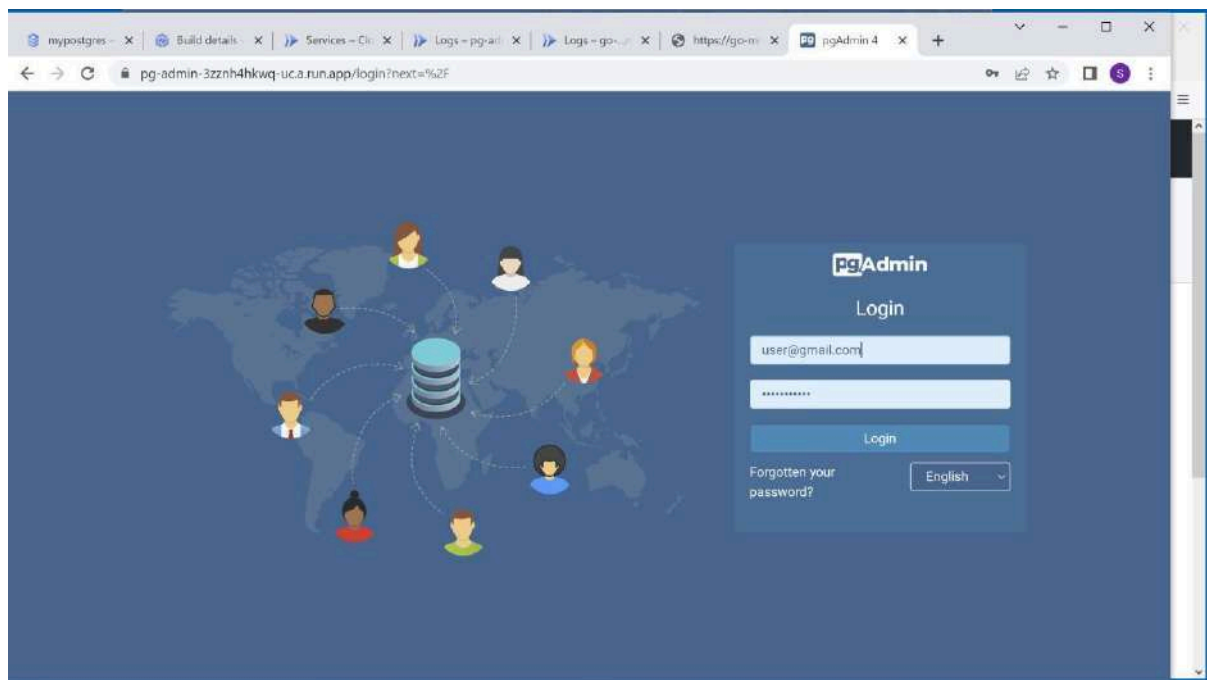


pgAdmin Microservice

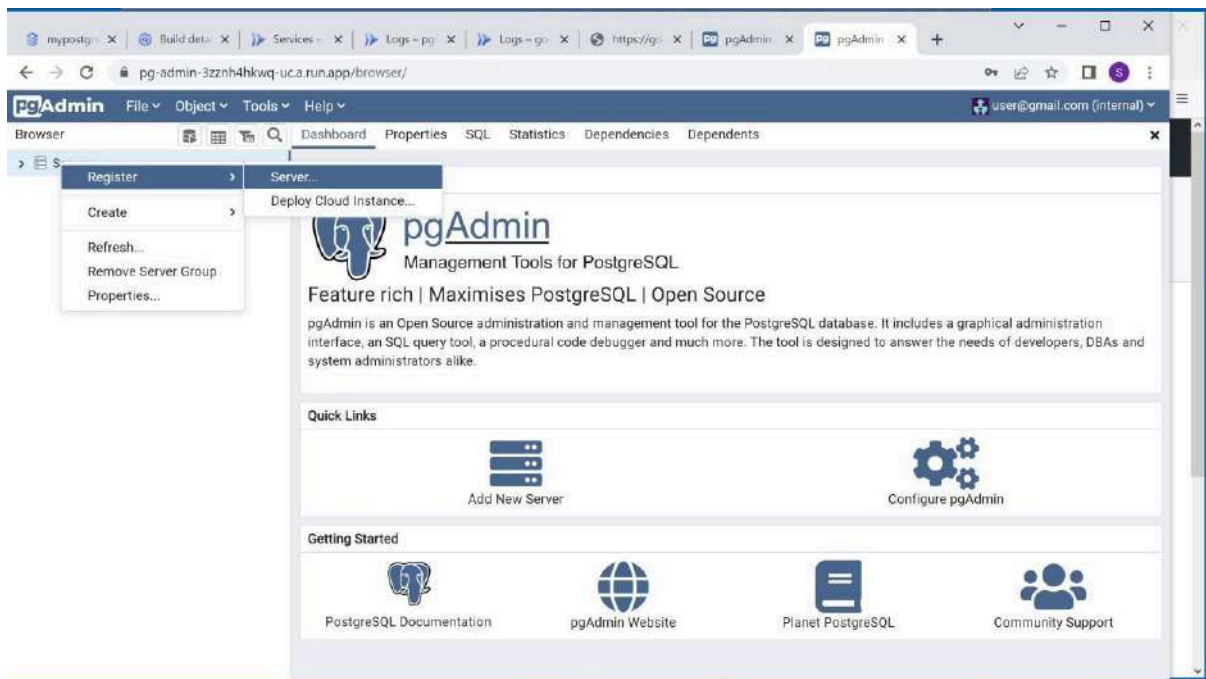
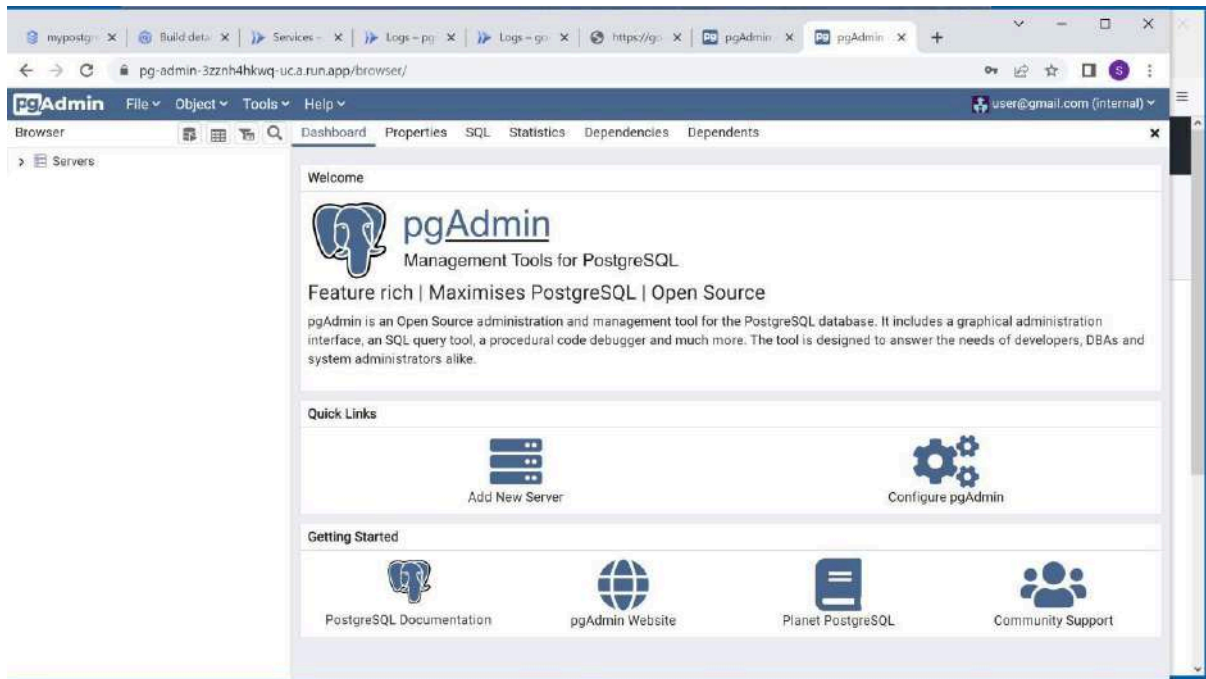
- From Cloud Run, click on pgadmin, copy the highlighted URL.



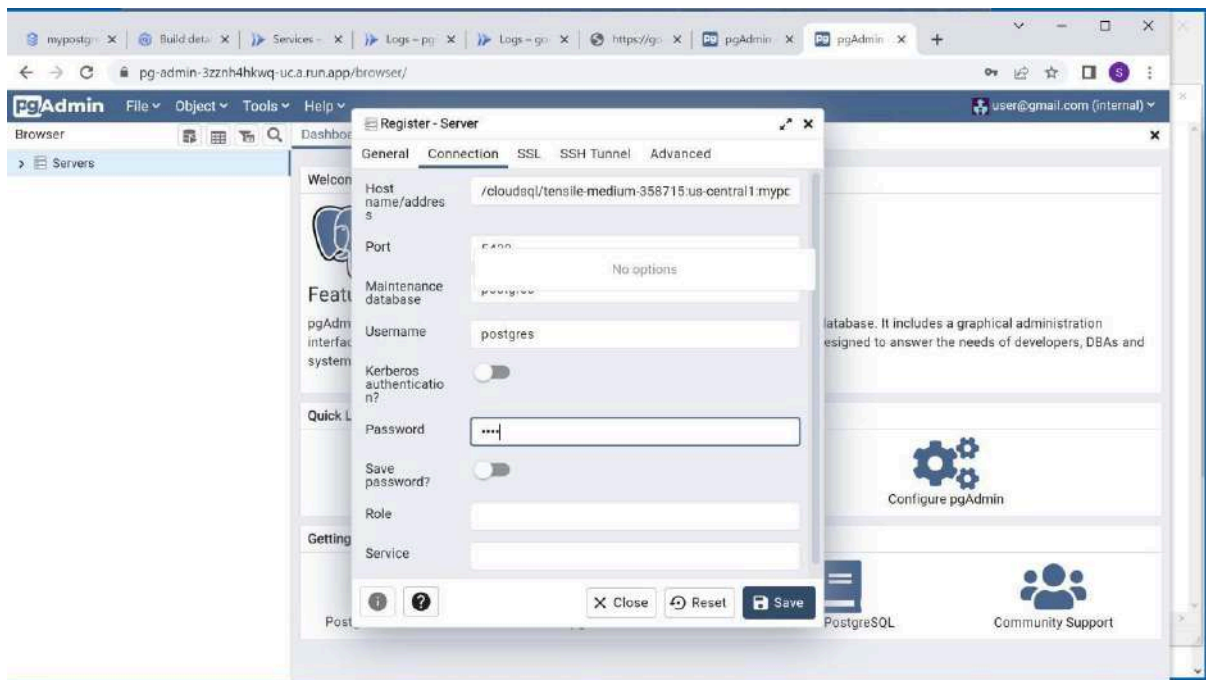
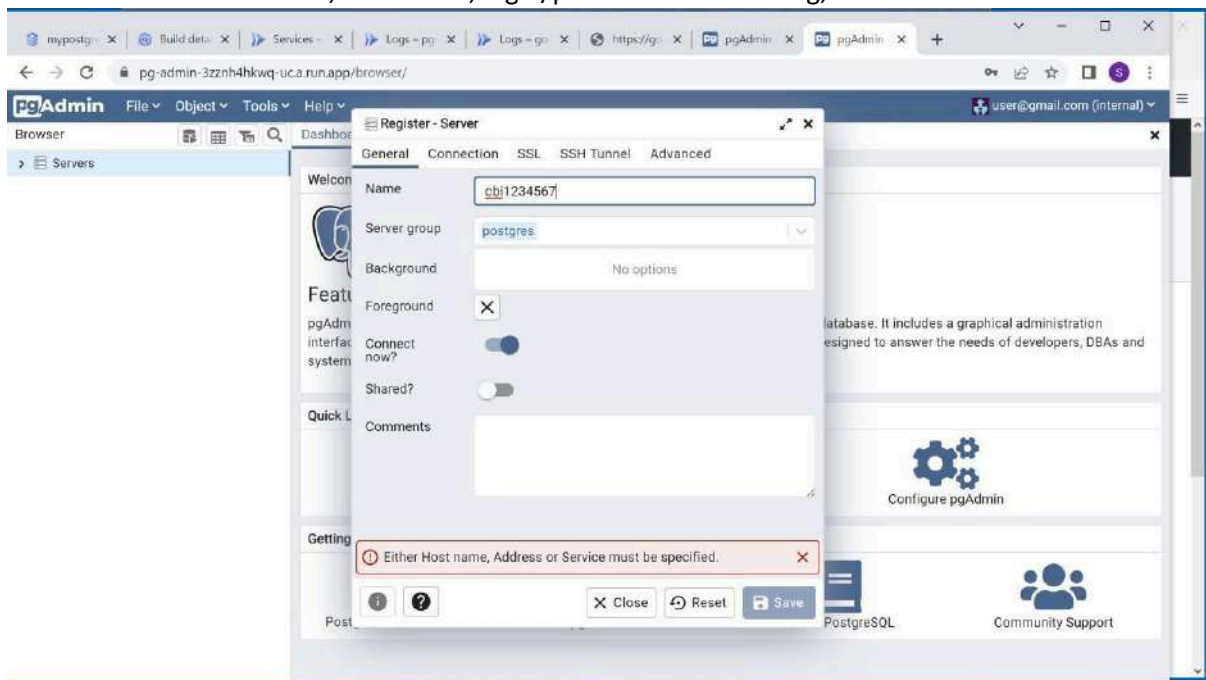
- Open the URL in a Browser and Login to pgadmin to validate that tables are created. Login credentials are in the cloudbuild.yaml file.



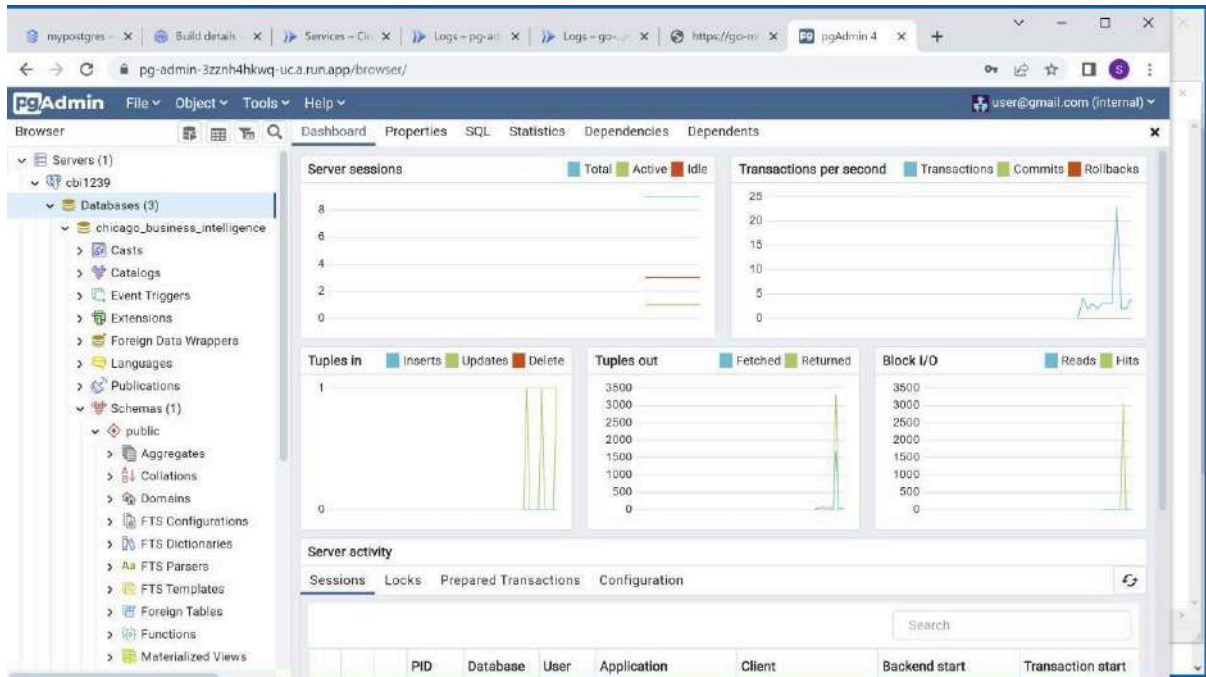
- Add a server.



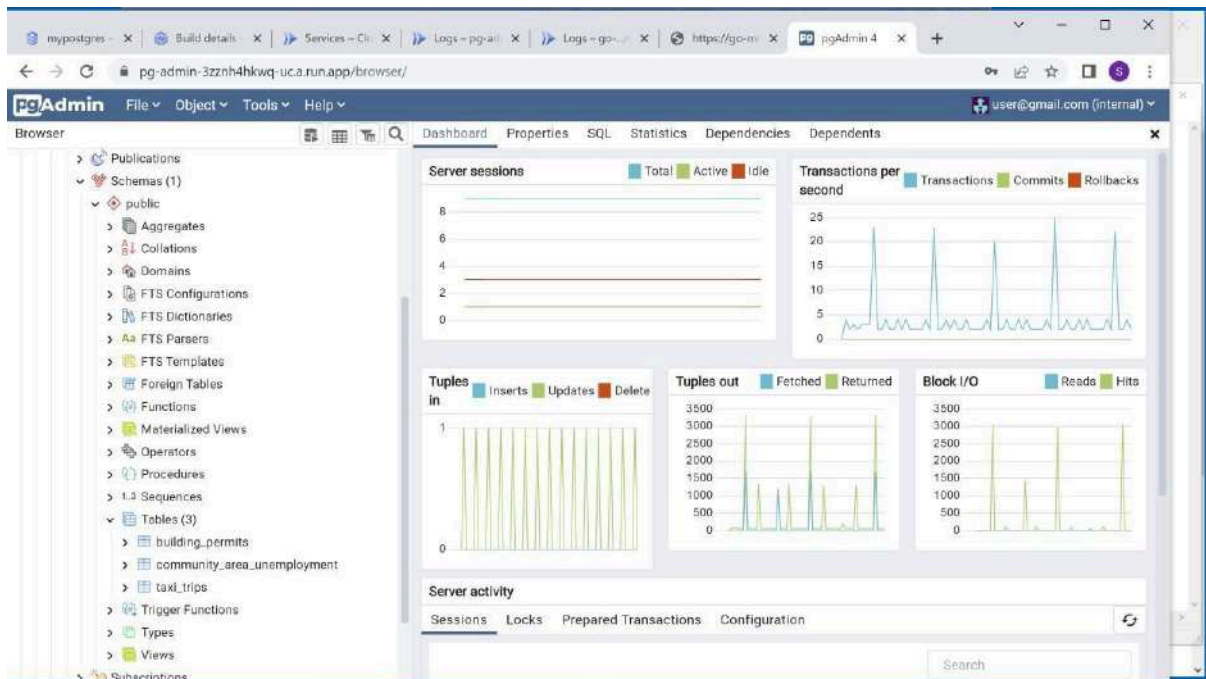
- Enter server name , host name, login/password in the dialog, click the SAVE button.



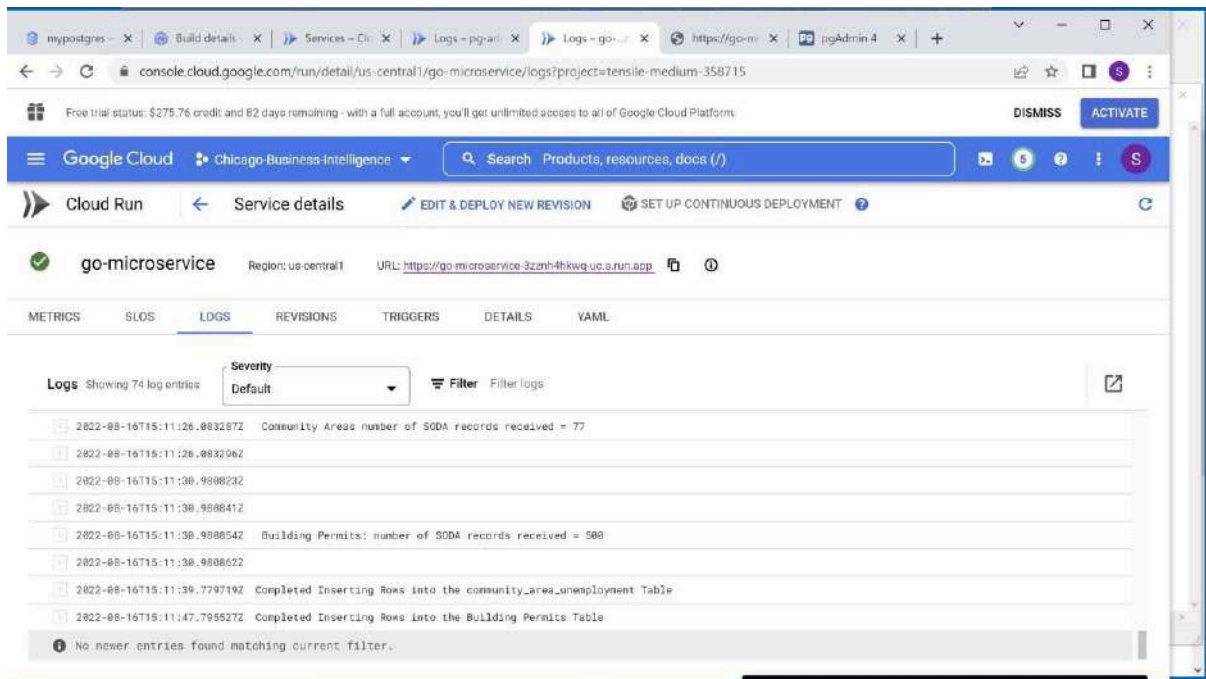
- After you login, click on Chicago_business intelligence.



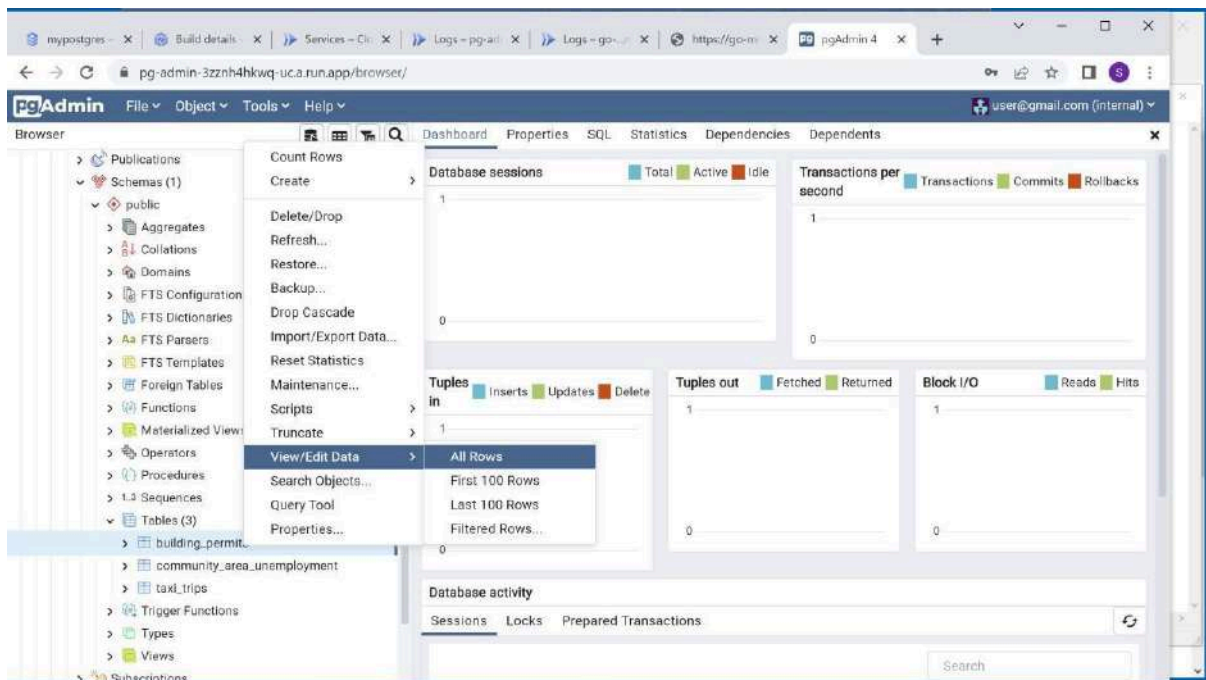
- Click on schemas/tables and verify you see the CBI tables



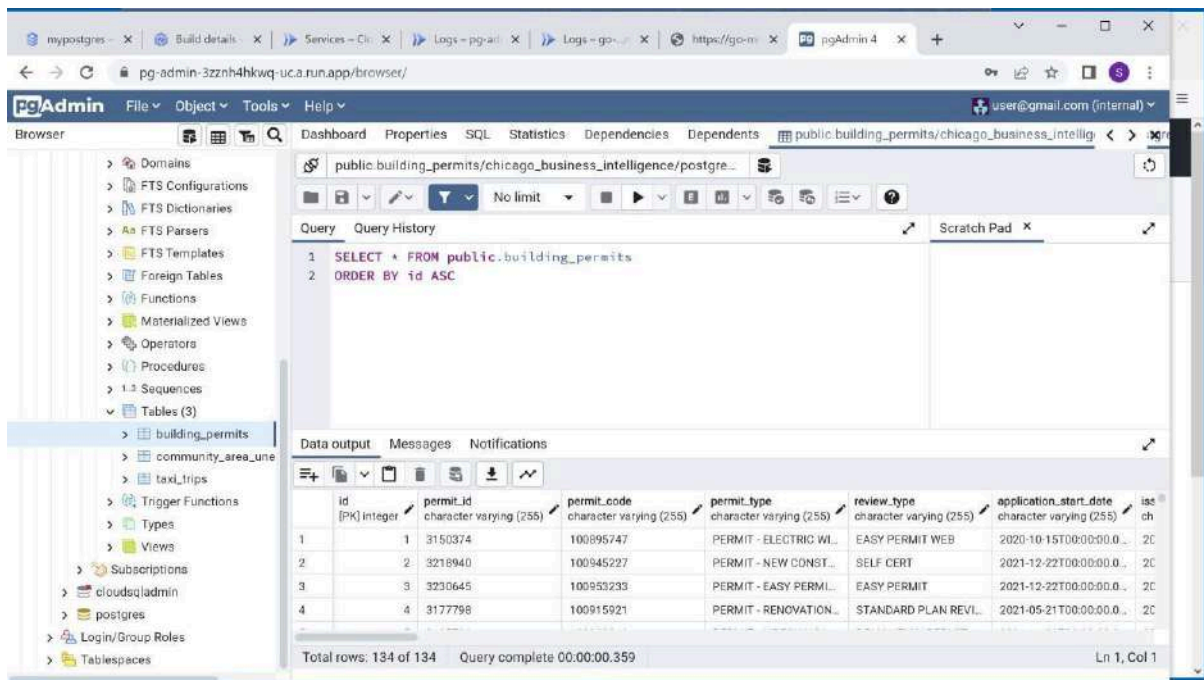
- Go to go-microservices and verify at least one of these tables got rows inserted into it.



- Go back to pgAdmin and select one of these tables and select view all rows:

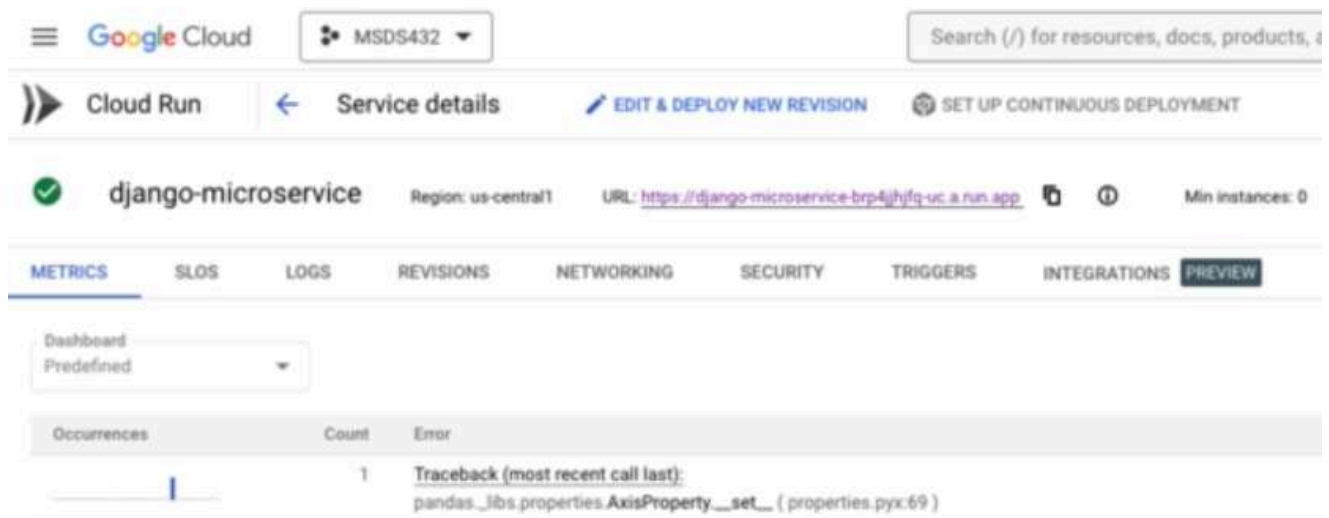


- Go back to pgAdmin and select one of these tables and select view all rows:



Django Microservice (Frontend)

- From Cloud Run, click on the Django Microservice, copy the highlighted URL to access the dashboard.



Welcome to the Chicago Business Intelligence Dashboards

- [Taxi Trips From Airport](#)
- [Taxi Trips with High CCVI](#)
- [Building Fee Per Year](#)
- [New Construction in Low Income Zip Codes Per Year](#)